A Data Mining-Driven Decision Support System

Hristo Borisov[1] and John Galletly[2]

[1] telerik, 33 Alexander Malinov Blvd., Sofia 1729, Bulgaria
hristo.borisov@telerik.com
[2] American University in Bulgaria, Blagoevgrad 2700, Bulgaria
jgalletly@aubg.bg

Abstract. The project is a data mining-driven decision support system that is designed to improve and assist managers in making rational and knowledgeable decisions. The system will make prediction of future e-mails overhead and forum replies, as well as identifying clusters of similar customers by analyzing the type of e-mails they send.

Keywords: data mining, business intelligence, decision making and support

1 Introduction

The application described here is a data mining-driven decision support system that is designed to improve and assist managers in making rational and knowledgeable decisions. The system mining structures and models are based on client-support material generated by a software company, which principally develops ASP.NET UI controls, obtained and used with permission from the company. The focus of the application is the prediction of future support e-mails, or forum posts, as well as identifying groups of customers and issues that most frequently appear in the support threads of the company.

Data extraction from the company's database, transformation, and loading into a data warehouse are performed with Microsoft SQL Integration Services. The prediction and clustering is then realized by the Time Series, Clustering and Associations algorithms provided by the Microsoft SQL Analysis Services against the data warehouse. These software components run on a server. The communication between the server and the client-side is via a Windows Communication Foundation (WCF) service.

The client-side of the application is a website that is based primarily on a Silverlight application which handles the user interaction and user interface. The interface consists of interactive charts and maps that help the user transfer data into knowledge and assists him in making rational decisions. The Silverlight application is built with Silverlight controls, and is implemented in accordance with Microsoft's Prism Composite Application Guidance which "enables the application to use loosely-coupled, independently evolvable pieces that work together in the overall application" [1]. The Composition Architecture of the project will enable requirements to be easily changed if and when new application requirements are introduced.

2 Application Functionality

The application provides the following functionality, as follows:

- Forecasting: Forecasting customer-support emails for a user-defined period; Filtering customer-support forecasts by type of support, e.g. forum, bug report, feature request, etc; Filtering customer-support forecasts by product or product team, e.g. Silverlight, WPF, Windows Forms.
- Associations: Recommending products to customers based on their already purchased items; Recommending controls or articles to customers based on their support experience; Finding associations among UI controls, and finding which UI controls are most probable to be used together, Identifying tangible measures of already existing incentive programs such as bonus points.
- Clustering: Identifying groups of customers that use specific type of controls together; Identifying the types of application and websites that users are creating – Line of Business, Entertainment, etc. and accessing and exporting the list of customers from the application.

3 Application Architecture and Design

The architecture of the system is composed of four integrated processing stages as illustrated in Fig. 1.

The Online Transaction Processing (OLTP) database is a store for the software company's transactional data such as support threads, messages, clients, products, etc. However, this data is neither structured nor appropriate for data mining since there is an excessive number of relationships and amount of data that is not needed for the requirements of the application. To overcome this, Extract, Transform, and Load (ETL) SQL statements were implemented and are applied to the data in order to create a data warehouse in which the mining algorithms operate. The ETL tasks are performed by Microsoft's SQL Integration Services [2] which extracts and loads the necessary information into the data warehouse.

The data warehouse itself is a Kimball snowflake-schema warehouse [3]. This has a fact table comprising all messages sent and received by the company in the course of its operations. Along with the fact table, a couple of dimensions must be defined that describe the facts in the table. For example, a typical dimension is a date, a location or a client. The data warehouse is the source for the mining structures and models that rely on clustering, times series, and association algorithms. Rather than implementing a complete data warehouse solution that stores information from all functional departments, the application's data warehouse is a single data mart which stores information only from the customer relationship department.

In order to explore patterns, associations, and relationships in the company's data, Microsoft's SQL Analysis Services [4] is used to create the mining structures and models. Three distinctive data mining algorithms are used to extract knowledge from the warehouse – the Microsoft Clustering Algorithm; the Microsoft Association Algorithm; and the Microsoft Time Series Algorithm. The mining structures and models associated with these algorithms are accessed by an instance of the SQL Analysis Services.

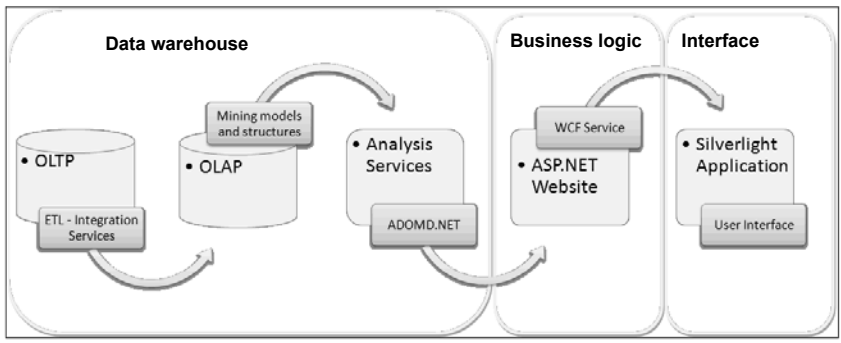

Fig. 1. Four integrated processing stages.

After the mining structures and models have been created, they are accessed by an ASP.NET website. An Analysis Management Objects (AMO) provider [5] in an ASP.NET application is used to access and process the mining structures and models in the Analysis Services. To browse and retrieve information from the mined structures, such as a prediction on time series, an ActiveX Data Object Multidimensional (ADO MD) provider [6] is used. The ADO MD provider executes Data Mining Extensions (DMX) statements against the Analysis Services to retrieve information such as a prediction on time series or a relationship among clusters. ADO MD uses the XML for Analysis protocol to transmit and receive SOAP requests and responses that are compliant with the XML for Analysis (XMLA) specification [7]. This is illustrated in Fig. 2.

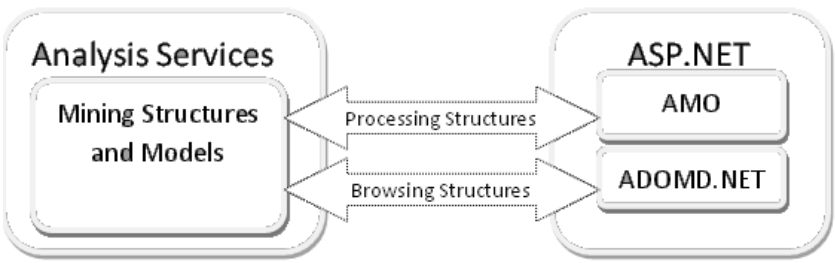

Fig. 2. Communication between Analysis Services and ASP.NET.

Since the results returned from the DMX queries are multidimensional in nature, an AdomdDataReader is used to flatten the results and create a List structure containing the necessary data.

The implementation of the AMO and ADO MD providers is incorporated into a single Windows Communication Foundation (WCF) [8] service that is asynchronously available to the client side by several public methods and data types. The communication between the server and the client side is composed of two parts: first, a WCF service together with the server side is hosted in an Internet Information Server (IIS), and second a WCF client is hosted in the client side that runs inside the user's browser. After information is available in the ASP.NET Web application, the WCF service is used to connect to the client-side Silverlight application and transfer the necessary data for display in the browser in the form of charts and maps.

4 Implementation of the Client Side

The user interface of the system is a single Silverlight application that is hosted inside the ASP.NET Web application. The communication between the server and the client will be done by the WCF service. The client side of the application is based upon the Prism Composition Application Guidance framework for Silverlight [1]. Prism is a small framework that has several assemblies allowing an application to be built from several loosely-coupled modules. The modules are integrated into a single application called the Shell. It displays all View parts of the integrated modules.

In order to benefit from the ability of Prism to unite several loosely-coupled modules into a single application, it is necessary to separate the functionalities of the application in different modules. Each module is an assembly that holds Views with similar functionality. There are total of five modules that are united into the Shell. In order to avoid redundancy of code, all interfaces, events and services are implemented in a single Infrastructure module that is referenced in each module and the Shell.

Below is a short description of each module:

- The Shell unites all modules into a single application that represents the UI Composition view. The Shell has reference to all modules.
- The Infrastructure module defines common classes and implementations used across the application such as events and services. Every module along with the Shell has reference to the Infrastructure assembly.
- The Navigation Module represents a CoverFlow UI control that enables the user to navigate among categories. After the selected category of the CoverFlow is changed, it publishes an Event that is received by modules that listen for this event. The payload of the event is the index of the selected category of the CoverFlow which is used by other modules for filtering.
- The Filtering Module encapsulates all filtering functionality. It has a single View that enables the user to filter the data of the Forecasting and Clustering modules. It has a single event that is published whenever the user clicks its Preview button.
- The Forecasting Module holds a collection of views that mainly use the WCF service to request data from the Time Series algorithm.
- The Clustering Module uses only data from the Association and Clustering algorithms to visualize reports such as tree maps.

5 Conclusion and Future Work

This paper has described the design and implementation of a novel data-mining decision support system that will allow senior managers to have better control of the company's customer support and predict changes for the future.

An extra application functionality to be added in the near future is Text Mining. This will identify the most frequent bugs and issues for a specific control by mining the support messages; identify the most frequently used keywords with the company tag in social networks; lookup specific keywords from social networks.

Many details of the implementation, as well as application screenshots, have been omitted from this paper because of page limit restrictions. Any reader who wishes to have further details should contact the authors.

Acknowledgment. The authors wish to thank telerik for their assistance in the preparation of this application.

References

1. Microsoft Best Patterns and Practices, Composite Application Guidance for WPF and Silverlight, October 2009, http://msdn.microsoft.com/en-us/library/dd458809.aspx
2. Microsoft - SQL Integration Services, http://www.microsoft.com/sqlserver/2008/en/us/integration.aspx
3. Snowflake Schema architecture, http://etl-tools.info/en/bi/datawarehouse_snowflake-schema.htm
4. Microsoft - SQL Analysis Services, http://msdn.microsoft.com/en-us/library/ms175609(SQL.90).aspx
5. Microsoft - Analysis Management Objects, http://msdn.microsoft.com/en-us/library/ms124924.aspx
6. Microsoft - ActiveX Data Objects (Multidimensional), http://msdn.microsoft.com/en-us/library/ms675532(VS.85).aspx
7. Microsoft - XML for Analysis Reference (XMLA), http://msdn.microsoft.com/en-us/library/ms186604.aspx
8. Microsoft - Windows Communication Framework, http://msdn.microsoft.com/en-us/netframework/aa663324.aspx